第四章：生成排列和组合

4.1 生成排列
4.2 排列中的逆序
4.3 生成组合
4.4 生成 *r* 子集

组合数学

(1) 存在：满足一定条件配置的存在性.

(2) 计数：计算出满足条件配置的数目.

(3) 算法：构造所有配置的算法.

(4) 优化：优化算法.

主要内容

- 排列生成算法：
  - 递归生成算法
  - 邻位对换算法
  - 从逆序生成排列算法
- 组合生成算法
  - 字典序
  - 反射Gray码
  - 基于字典序的r组合生成算法

第四章：生成排列和组合

4.1 生成排列

4.2 排列中的逆序

4.3 生成组合

4.4 生成 *r* 子集

一种最为初级的“黑客”技术

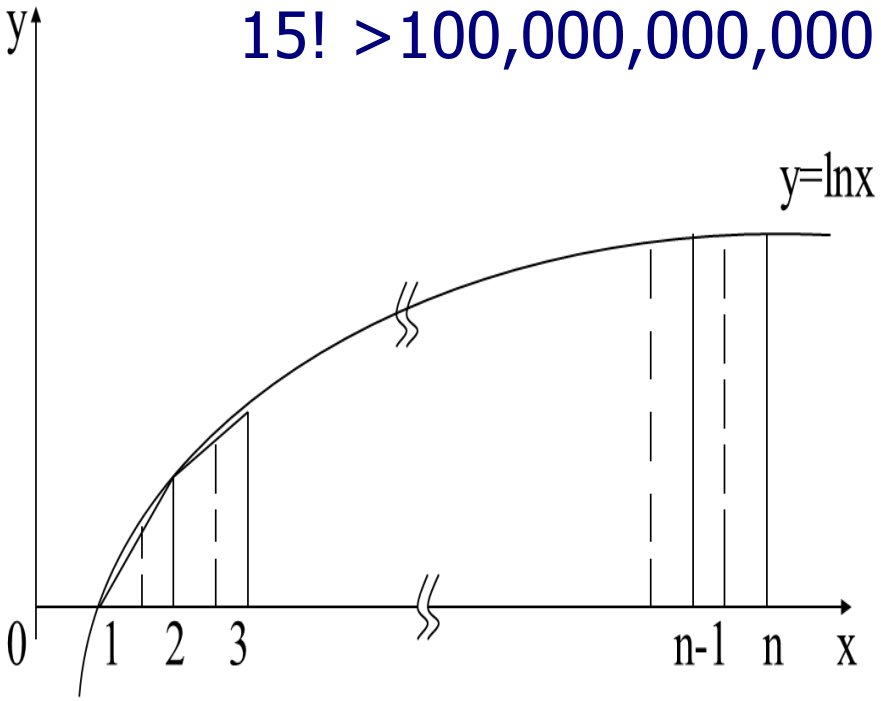
- **穷举攻击**：最初的**DES**密码是**40**位二进制数。编一个程序，尝试所有可能的密码。要求：
  - 无重复、无遗漏
  - 尽量少的存储空间
  - 尽可能简单操作。
- 如果具有一些“预先知识”，在一些特点字符里选取，如何设计算法？（如字典攻击）

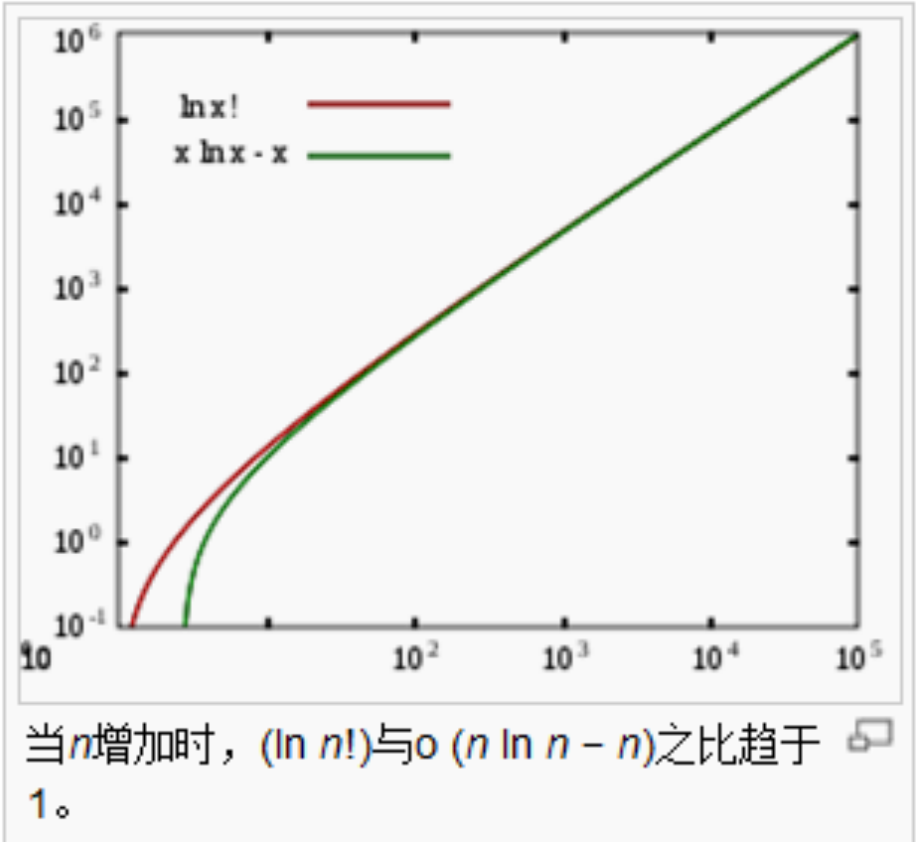
4.1 生成排列

- **Stirling近似公式**

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

15! >100,000,000,000





排列生成算法

- 生成{1, 2,…, *n*}的所有排列的算法
  - 算法输出结果为一个表
  - 表包含了{1, 2, …, *n*}的所有排列
  - 每个排列只出现一次

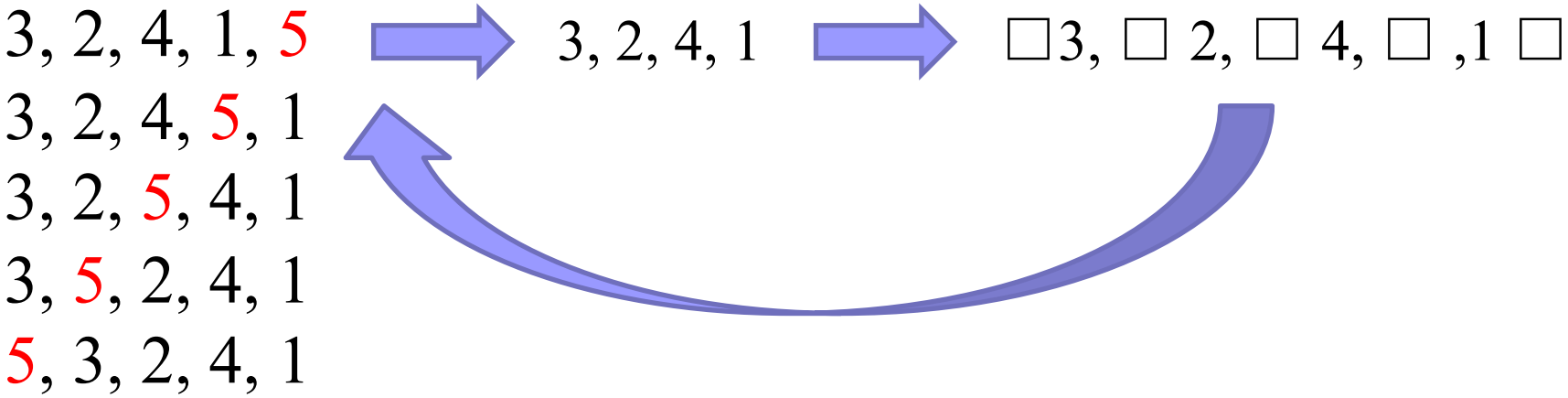
三种排列生成算法

- 递归生成算法
- 邻位对换算法
- 从逆序生成排列算法

递归生成算法

- **S.M.Johnson (1963)/ H.F.Trotter(1962)**

{1，2，3，4，5}　　{1，2，3，4}



- 观察1：将整数 n 从{1, 2, …, n}的一个排列中删除后，得到一个{1, 2,…, n–1}的排列。
- 观察2：同一个n-1排列可以从不同的（n个）n排列生成
- 观察3：从{1, 2,…, n–1}的一个排列可生成 n 个{1,2,…,n}的排列

算法基本思想

- 对集合{1, 2, …, *n*-1}的每一个排列进行如下操作（一共(*n*-1)!个排列）：
  - 把 *n* 插入到首、尾和任两个数的中间共 *n* 个位置，产生集合{1, 2, …, *n*}的 *n* 个排列
- 从而产生 *n*×(*n*-1)!=*n*!个集合{1, 2, …, *n*}的排列。

n=5时　□3, □ 2, □ 4, □ ,1 □

{1}的所有排列 → {1, 2}的所有排列 → {1, 2, 3}的所有排列 → … → {1,2,…,*n*}的所有排列

算法描述

n=1: 1

n=2: 1 2

2 1

n=3: 1 2 3

1 3 2

3 1 2

3 2 1

2 3 1

2 1 3

n=4: 1 2 3 4

1 2 4 3

1 4 2 3

4 1 2 3

…

2 3 1 4

2 3 4 1

2 4 3 1

4 2 3 1

4 2 1 3

2 4 1 3

2 1 4 3

2 1 3 4

n=5, …

当生成的排列为213…*n*时，算法结束，生成全部排列。

算法分析

- 可归纳验证该算法生成的最后一个排列是213…n。

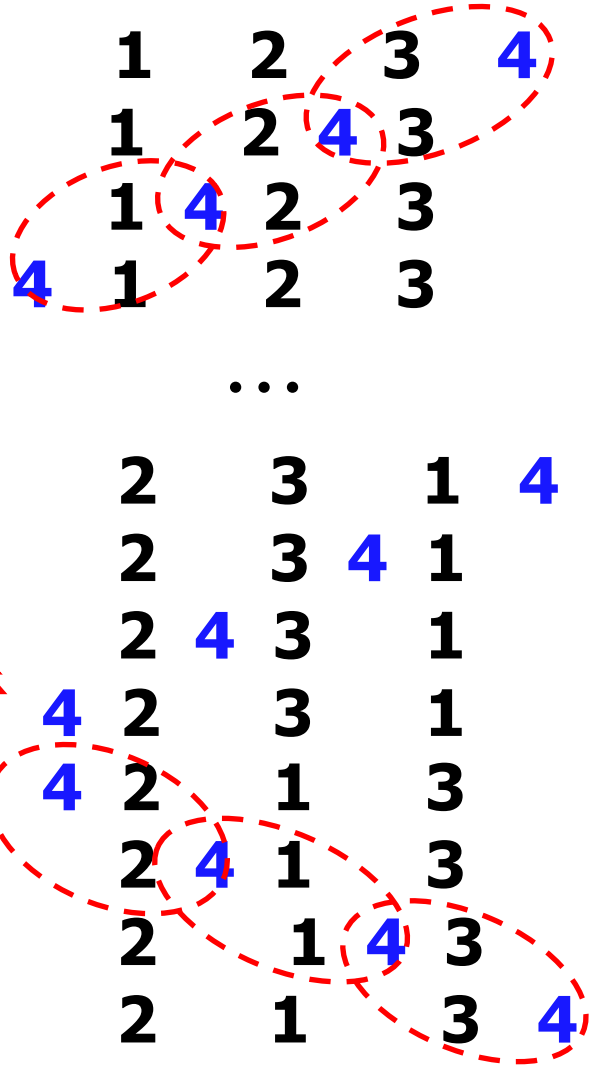
算法特点：

- 生成{1, 2, …, n }的排列算法需要存储所有 {1, 2, …, n-1}的排列，因此，需要巨大的存储空间。
- 算法空间复杂度太高！

观察：

注意：交换两个相邻的数

n=4:



什么条件下进行邻位互换？

邻位对换算法

- 对任一给定整数 k, 其上加一个箭头表示移动方向：$\vec{k}$ 或 $\overleftarrow{k}$ 。
- 对于集合{1, 2, …, n}的任一个排列，其中每一个整数都有一个箭头指出其移动方向，

如果整数 k 的箭头指向与其相邻但比它小的整数, 则称 k 是可移动（活动）的.

例：序列 $\vec{2}\,\vec{6}\,\vec{3}\,\overleftarrow{1}\,\vec{5}\,\vec{4}$ 那几位是活动的？

只有3、5、6是活动的。

注：(1) 在任意序列中，1 绝对不可能是活动，是否正确？ 正确！

(2) 在{1, 2, …, *n*}元素构成的任意序列中，*n* 是否一定是活动的？

除去以下两种情况：

- *n*是第一个整数而它的箭头 指向左边：

$$\overleftarrow{n}\ldots$$

- *n*是最后一个整数而它的箭头指向右边：

$$\ldots\ \overrightarrow{n}$$

邻位对换算法

生成{1, 2, …, n}的排列算法:

1. 初始：$\overleftarrow{1}\ \overleftarrow{2}\ \ldots\ \overleftarrow{n}$;

2. while 存在活动整数时， do

   (1) 求出最大的活动整数 *m*

   (2) 交换 *m* 和其箭头指向的相邻整数的位置

   (3) 改变所有满足*p*>*m*的整数 *p* 的箭头方向。

3. 不存在活动整数时，算法结束。

算法例子

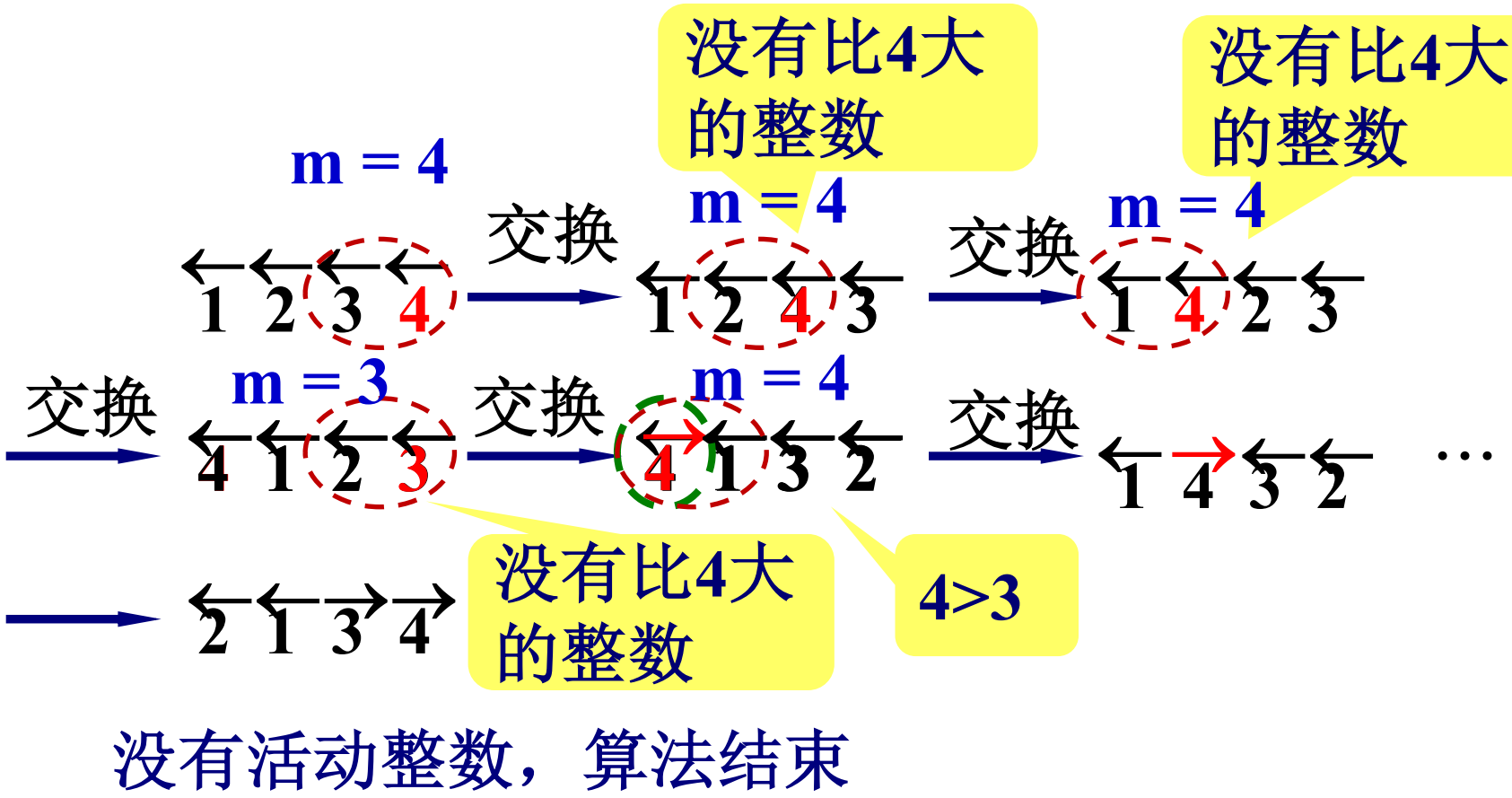
2. 存在活动整数时，do

(1). 求出最大的活动整数m

(2). 交换m和其箭头指向的相邻整数的位置

(3). 改变所有满足p>m的整数 p 的箭头方向

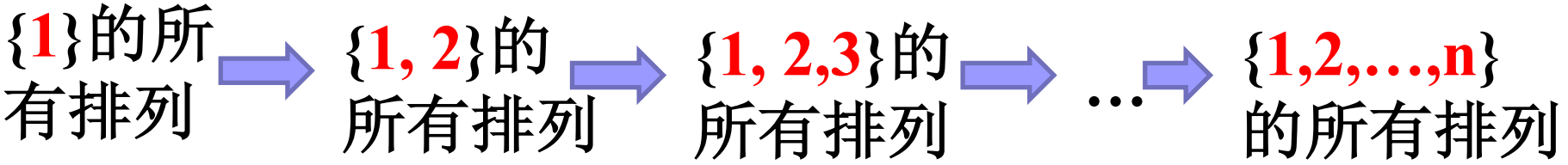
n=4的算法描述



没有活动整数，算法结束

排列生成算法

- 递归生成



- 邻位对换算法
  - 从排列 123…*n* 开始，生成所有 *n* 阶排列
  - 活动：箭头指向比其小的整数
  - 邻位对换
- 结论：两种算法生成的排列顺序一致